REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY (*LEAVE BLANK*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	14 June 1996	Professional Paper

4. TITLE AND SUBTITLE

Using an Ordnance Server to Provide Validated Weapon Models to MODSAF

5. FUNDING NUMBERS

6. AUTHOR(S)

Larry Ullom

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Commander
Naval Air Warfare Center Aircraft Division
22541 Millstone Road
Patuxent River, Maryland 20670-5304

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Naval Air Systems Command
Department of the Navy
1421 Jefferson Davis Highway
Arlington, VA 22243

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

The Modular Semi-Automated Forces (MODSAF) simulation tool traditionally models a weapon's physical and behavioral characteristics by loading simplistic algorithms that are derived from generic data into the main simulation application. This causes inherent problems. The processor load of the weapon models must remain relatively low to maximize the entity count per workstation. This is often the leading cost driver in an exercise. In addition, using sensitive weapon data imposes security issues on the MODSAF application.

14. SUBJECT TERMS	15. NUMBER OF PAGES
MODSAF, Modular Semi-Automated Forces;MFS;ACETEF	16
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18

Enclosure (9)

Using An Ordnance Server to Provide Validated Weapon Models to ModSAF[†]

Larry Ullom
Aircraft Simulation Branch
Naval Air Warfare Center Aircraft Division
48140 Standley Road
Patuxent River, MD 20670-5304
lullom@msis.dmso.mil

Peter Fischer
J.F. Taylor Inc.
P.O. Box 760
Lexington Park, MD 20653
pfischer@jfti.com

1. Abstract

The Modular Semi-Automated Forces (ModSAF) simulation tool traditionally models a weapon's physical and behavioral characteristics by loading simplistic algorithms that are derived from generic data into the main simulation application. This causes inherent problems. The processor load of the weapon models must remain relatively low to maximize the entity count per workstation. This is often the leading cost driver in an exercise. In addition, using sensitive weapon data imposes security issues on the ModSAF application.

The operational requirements for certain training exercises dictate that weapon models must perform as a functionally valid replica of the actual system. This ensures the training performed would enhance the trainee's performance in a similar situation. Given the current approach, it is too computationally expensive to add validated models to the actual ModSAF application. The use of an Ordnance Server (OS), as demonstrated by the Air Combat Environment Test and Evaluation Facilities' Manned Flight Simulator (ACETEF/MFS), provides a better solution.

The Ordnance Server is an external host that models weapons surrogates. Validated weapon models are incorporated into the Ordnance Server and the corresponding ModSAF models are disabled. This approach improves scalability, provides a more level playing field between interacting entities, and segregates sensitive or classified modeling and data.

This paper will discuss the origin of the Ordnance Server concept and the process of integrating an Ordnance Server with ModSAF. An analysis of the test implementations will show the benefits of this approach. The paper will also include a discussion of open issues such as in flight guidance input from the launching entity. Finally a conclusion that looks ahead to future implementations will be provided.

2. Introduction

Traditionally the modeling of a weapon's physical and behavior characteristics was done in the model that was responsible for launching the weapon. This arrangement was the natural environment in systems that were designed to execute as a monolithic simulation on a single host. This type of system is highly limited in a number of important areas including scalability and multiplicity of reuse.

The ModSAF system was designed to address some of the problems in monolithic computer generated forces (CGF) systems. It uses distributed interactive simulation (DIS) protocols to allow multiple simulation hosts to cooperatively interact in a unified synthetic battlespace. However the concept of retaining ownership of all simulation attributes spawned by the ModSAF application was retained.

To further enhance the capabilities of CGF systems such as ModSAF, it is necessary to allow hand-off of simulation entities and attributes to other more specialized simulators. While this is explicitly addressed in the new Department of Defense (DoD) High Level Architecture (HLA), it is also possible to gain the benefits of using validated munition models with ModSAF in a DIS environment. The use of an Ordnance Server can provide this capability.

3. Technical Overview

The Ordnance Server extends the idea of distributed simulation by separating the simulation of the launching vehicle from the munition simulation. The concept could be applied to any pairing of munition and launch vehicle simulations. However this paper shall consider the ModSAF launcher only. In order to understand how the Ordnance Server can be used with ModSAF it is necessary to first discuss it's internal workings.

The Ordnance Server operates using only standard DIS protocol data units (PDUs). When a cooperating launch vehicle simulation wishes to fire a munition, it issues a fire PDU as it normally would. The Ordnance Server, having been previously configured to look for fire PDUs from a specific (site, application, entity), will try to match the weapon type and fusing data to a weapon it is configured to simulate. If a match is found, the Ordnance Server will instantiate a simulation of that weapon using target data from the fire PDU. The Ordnance Server issues entity state PDUs for the instantiated munition during it's delivery to the target. When the fuse model indicates the termination of the munition, the Ordnance Server generates a detonation PDU.

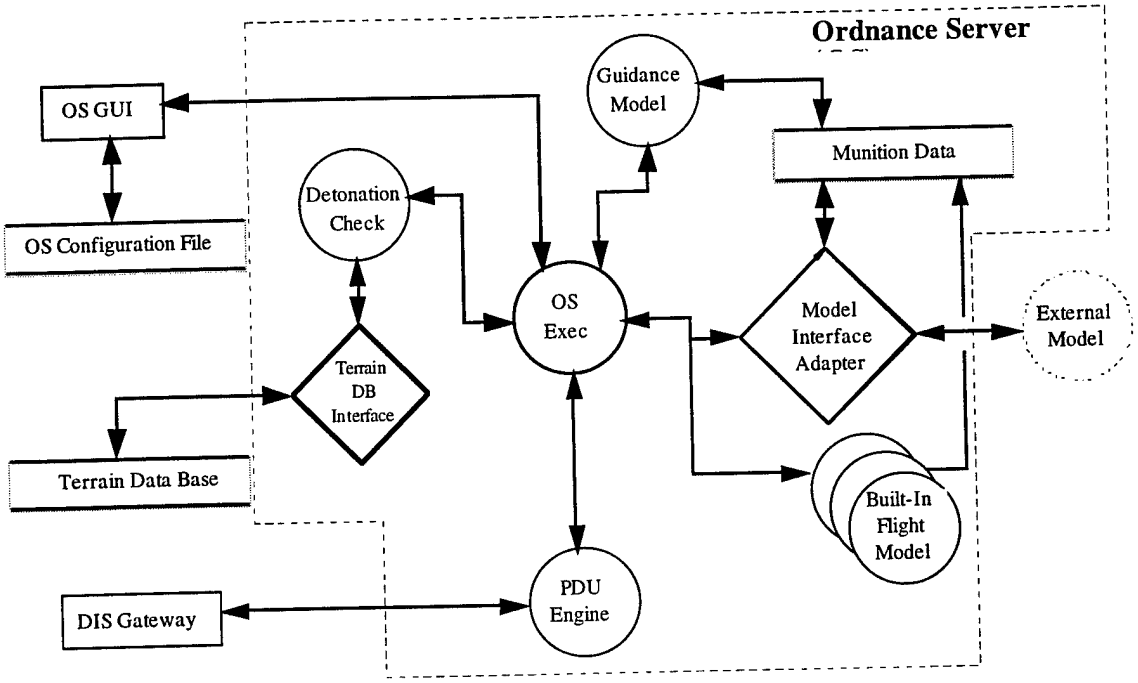


Figure 1

3.1 Special Interfaces

The Ordnance Server supports two interfaces to external objects. These interfaces and there relationship to the rest of the Ordnance Server is illustrated in figure 1. The first is a Model Interface Adapter (MIA). This is a code wrapper that goes around an external weapon model and provides all the translation services needed to make the model look like an internal simulation to the server executive. At the same time, the MIA simulates the environment the external model was designed to operate in. This architecture provides a mechanism for interfacing nearly any legacy model. In particular, validated models from accepted training systems can be used to support a DIS exercise.

The second interface is the ground truth database interpreter. This provides the OS with a consistent representation of ground truth regardless of the actual format used in the underlying terrain database. The only parameter that is important to most weapon models is the height above the terrain at the point where the munition is currently flying. Other parameters could be included in this interface if a particular model required them for normal operation.

3.2 Configuration Parameters

The Ordnance Server must be configured to operate with a "parent" launch entity. The parent entity is denoted by the (site, application, entity) triplet. The Ordnance Server can serve multiple hosts by allowing the entity or entity/application identifiers to be wild cards. The entity type of any munitions to be surrogated must also be specified and mapped to a specific weapon model. Any weapon model loaded can be mapped to any munition type. Other configuration information that must be supplied includes the terrain database file name, DIS exercise and IP numbers, and types of runtime feedback desired. There are model specific parameters associated with each weapon simulation as well.

4.0 Integrating ModSAF with the OS

Integration with the Ordnance Server first required that the ModSAF internal weapon dynamics models be disabled. This resulted in relatively simple modifications because of ModSAF's highly modular design. To suppress entity state and detonation PDUs, libmissile was modified so ModSAF would only issue the Fire PDU. This allows the Ordnance Server to describe the trajectory and detonation event of the munition. A command line switch "Generate Missiles" was added to the ModSAF executable. When "-nomissiles" is specified, ModSAF's missile simulations are suppressed. Also, libmlauncher was modified to removed the munitions ID number from an internal list of local entities in ModSAF. Without the id number in it's local list, ModSAF will recognize the Ordnance Server's missile as a viable remote entity. Otherwise ModSAF would still act as if the entity were local, resulting in no icon display for the munition on the Plan View Display.

Another problem was that the designated target id was not originally specified in the Fire PDUs produced by ModSAF. This information is needed for the Ordnance Server to work correctly. The intended target was originally kept internally in ModSAF, so the change consisted of simply passing this data to the routine used to broadcast ModSAF's Fire PDUs.

Next a ground truth database interpreter based on the ModSAF libctdb services was added to the Ordnance Server. This allows both applications to share identical representations of ground truth.

4.1 Ordnance Server Advantages

The ordnance server can be used to provide models, that are accepted by the subject mater experts in a given exercise, with no penalty to the ModSAF application's processor load. These accredited models can be classified (if required) without affecting ModSAF's unclassified status. Maintaining multiple models for a particular munition at different levels of detail or different classification levels is also facilitated by this approach. Additionally a well designed network topology can reduce latency impacts on entity interactions, by collocating ordnance servers with the targets they are likely to engage.

The ordnance server has successfully been used to complement CGF systems in many large scale exercises. These exercises include the Strategic Theater of War (STOW) Engineering Demos, Navy Kernel Blitz fleet training event, and I/ITSEC 95 DIS Demo. In it's earliest use, a single ordnance server was used by one site to simulate only Air-to-Air missiles initiated by a single man in the loop system. Since then, many additional validated models have been added to the ordnance server and CGF systems have been modified to allow the ordnance server to simulate their munitions. As more applications use the ordnance server to simulate their munitions, the benefits to the goal of a fair fight become more apparent. Simulations that use the ordnance server will not create munition simulations with unrealistic flight or guidance attributes.

Due to the large number of entities simulated in a STOW exercise, the ordnance server's ability to further distribute processing load is especially useful and even necessary if high fidelity, real time munition models are an exercise requirement. The flexibility of the manner in which the ordnance server can be used to do this has been demonstrated. A local ordnance server has been used to simulate munitions launched by an application located at a remote site. Multiple ordnance servers have been used by one

application, each one simulating different types of munitions for the same set of entities. This characteristic of being flexible to the needs of a particular exercise scenario or hardware configuration has proven to be especially useful due to the variant nature of DIS exercises.

4.2 Open Issues

There are still open issues to be resolved with this approach. One of the major concerns regards tightly coupled systems, where the launch platform and the munition depend on either a one way or bi-directional link to function properly. An example is the Navy's SM-2 which receives steering input throughout flight from the launcher based on the launcher's radar track.

One solution to this would be to implement the link data via signal PDUs. This would increase the fidelity of the simulation. However it would also cost network bandwidth. Another solution would be to locate the sensor model on the Ordnance server. It is not clear how this could be facilitated under current DIS protocols, but the HLA fully supports this method.

Another issue is the loading of prelaunch data from the launching platform into the weapon model. Currently this is accomplished via the graphical user interface (GUI). The signal PDU is not a natural choice for this data as it would be passed via internal busses on the actual platform. One possible solution is the set data simulation management PDU. It could be used to initialize a weapon that requires this type of data prior to launch then the weapon would simply be attached to the launch platform until the fire PDU. This would require more extensive modification of ModSAF to support.

5. Conclusions

The ordnance server approach answers the problem of providing validated models for ModSAF by using an external host to take over the flyout of weapons launched by the application to the intended target. This allows ModSAF to use appropriate weapon models for the exercise event with no additional configuration management or piecemeal code integration. The models integrated in this manner do not have to be reengineered to fit within the ModSAF architecture, and multiple models of the same munition can be substituted easily.

The concept of the use of an ordnance server to supplement CGF applications has evolved into a tested, working product. The benefits of such an approach have been demonstrated through multiple DIS exercises. As more munition models are added to the ordnance server and the existing models capabilities are further refined, the munition simulations for all applications that use the ordnance server are improved.

Further research should provide answers to the open issues discussed in this paper. It is apparent that the advantages of this solution warrant continued development.

6. Acknowledgment

The authors wish to thank the following people for their contributions to this effort: CDR. Peggy Feldmann, Brett Dufault, John DiCola, Alexandra Wachter, and David Mutschler.

7. References

DiCola, J, Fischer, P, Mutschler, D, Ullom, L (1996) "Improving Munition Simulation Fidelity Through Use Of An Ordnance Server", *Proceedings of the AIAA Flight Simulation Technologies Conference* (to be published)

DiCola, J, Mutschler, D, Ullom, L, Wachter, A (1996) "Providing Common Munitions Models Via an Ordnance Server" *Proceedings of the 18th Interservice Industry Training Systems and Education Conference* (to be published)

Foster, L, Feldmann, P (1995) "The Limitations of Behavior for Valid Distributed Interactive Simulation" *95-13-027 Position Papers of the 13th DIS Workshop on Standards for the Interoperability of Distributed Simulations*

8. Biographies

Lawrence C. Ullom is a Senior Electronics Engineer with the US Navy's Naval Air Warfare Center Aircraft Division. He holds a BSEE from West Virginia Institute of Technology. He has been involved in numerous DIS working groups and demonstration projects. His interests are Networking, Simulation, and Distributed Processing.

Peter Fischer is an Electrical Engineer employed by J.F. Taylor, Inc. and working on site at the ACETEF/Manned Flight Simulator in the Naval Air Warfare Center at Patuxent River, MD. He received his bachelor's degree from George Mason University and is currently pursuing a master's degree in Computer Science from the Florida Institute of Technology. His interests include simulation, artificial intelligence, and computer graphics.

USING AN ORDNANCE SERVER TO PROVIDE VALIDATED WEAPON MODELS TO MODSAF

Authors:

Lawrence Ullom
NAWCAD Patuxent River

Peter Fischer
J.F. Taylor Inc.

Introduction

- Naval Aviation Subject Mater Experts Question Outcome of ModSAF Engagements
- Requires Use of "Accredited" Models for Training Exercise
- TACTS Models Available, but Not Easily Placed in ModSAF
- Previous Use of Ordnance Server with Manned Simulators Seemed Applicable

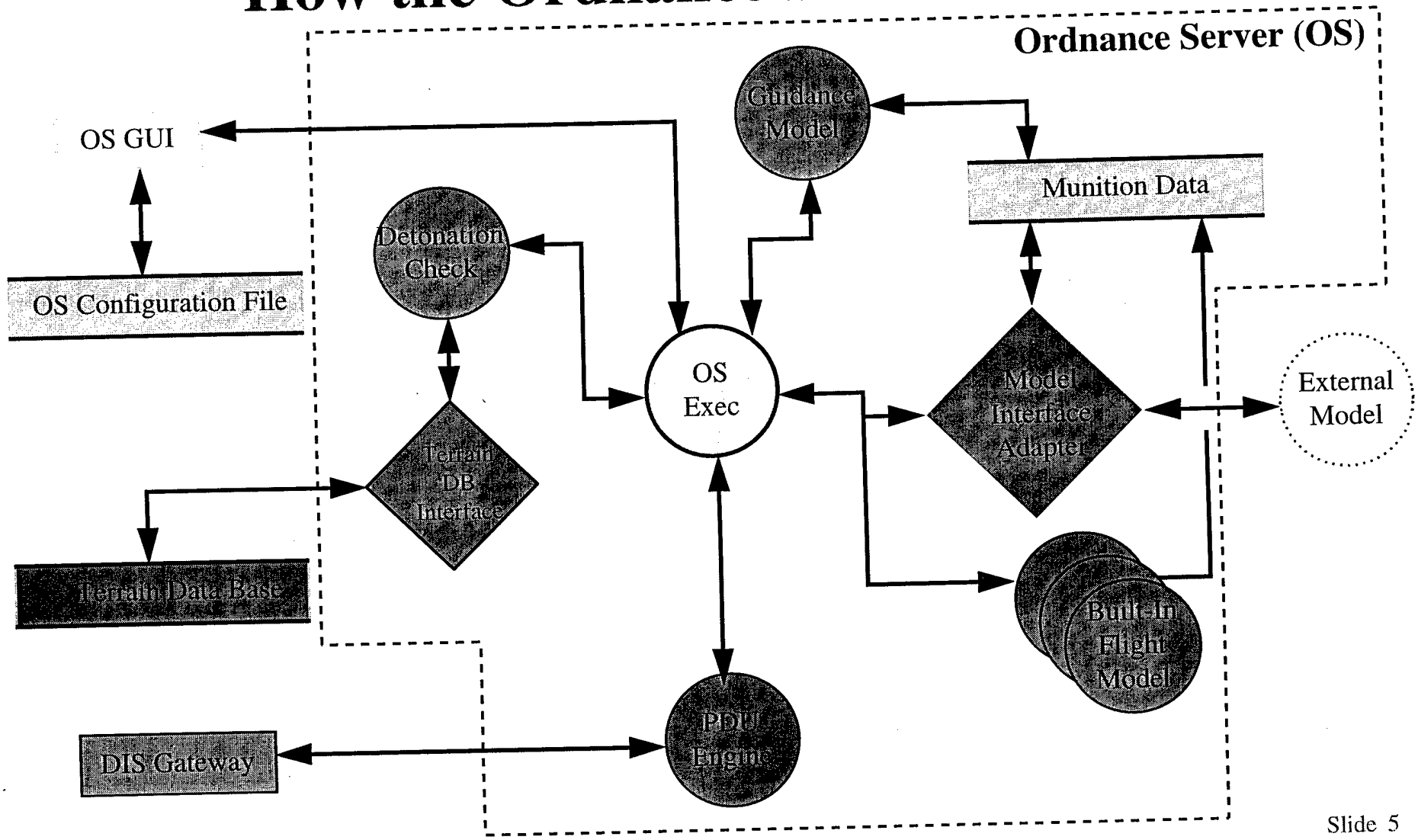
Perceived Problems With ModSAF Weapons

- Unrealistic Effective Ranges
- Susceptibility to Network Load
- Inaccuracy of Seeker Model
- Immunity to "Spoofing" Tactics
- Simplistic Hit / Kill Tables

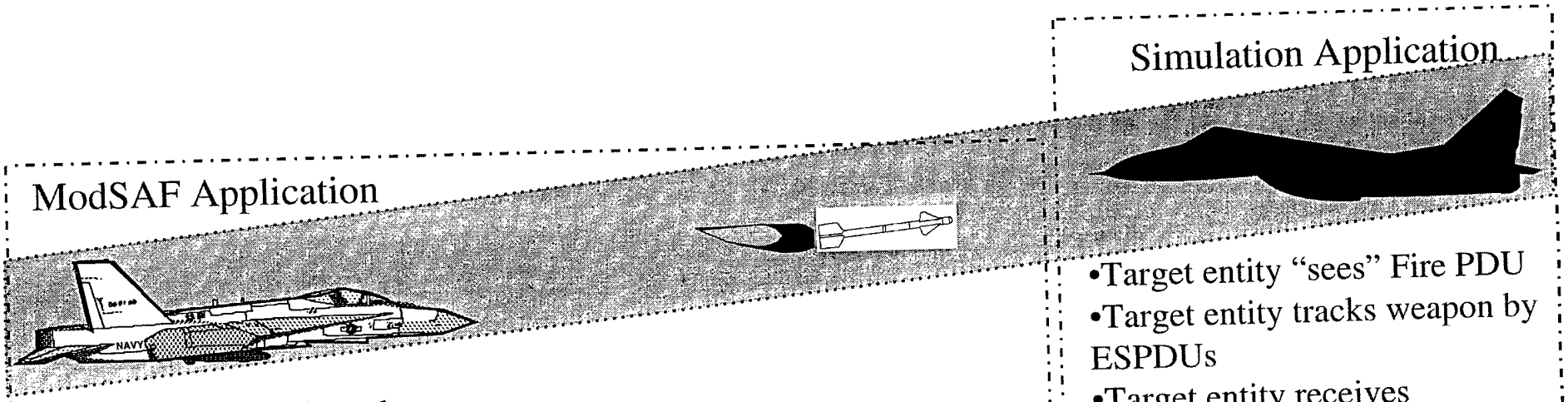
What Are The TACTS Models

- Navy - TACTICAL AIR COMBAT TRAINING SYSTEM | Air Force - ACMI
- Live Range Training Simulations Used to Resolve the Outcome of Combat Games
- Accepted (Read Accredited) by Aviators as "Good Enough"

How the Ordnance Server Works



Conventional DIS Weapon Simulation

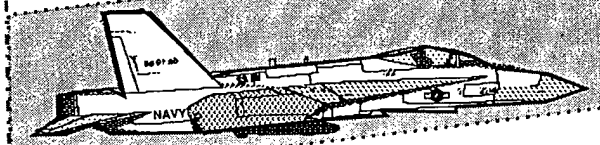


- Launching entity launches weapon with Fire PDU
- Launching entity sends ESPDUs of munition flyout
- Launching entity determines detonation result
- Launching entity broadcasts Detonation PDU

- Target entity "sees" Fire PDU
- Target entity tracks weapon by ESPDUs
- Target entity receives Detonation PDU
- Target entity performs battle damage assessment

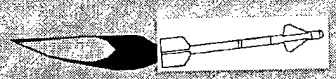
Weapon Simulation With An Ordnance Server

ModSAF Application



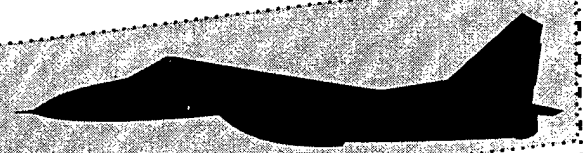
- Launching entity launches weapon with a Fire PDU

Ordnance Server



- OS receives Fire PDU
- OS performs munition flyout and issues ESPDUs
- OS determines detonation result
- OS broadcasts Detonation PDU

Simulation Application



- Target entity "sees" Fire PDU
- Target entity tracks weapon by ESPDUs
- Target entity receives Detonation PDU
- Target entity performs battle damage assessment

Pertinent Fire PDU Determination

- Pre StartX OS Configured With
  - "Parent" Site | Application | Entity
  - Entity Types for "Known" Weapons
  - Model to Use for Entity Type
- Post StartX OS Applies Filters to Fire PDU Based on Configuration Data
- Incomplete Fire PDUs Discarded

Integrating the Ordnance Server With ModSAF

- Add Target ID to ModSAF generated Fire PDU
- Disable Entity State (ES) PDUs and Detonation PDU in *libmissile*
- Modified *libmlauncher* to Remove Munition from Local List
- Added -nomissiles Command Line Switch to Allow Normal or OS Behavior

Conclusion

- Ordnance Server Offers:
  - Independent Munition Model Development
  - Reuse of "Proven" Models
  - More Level Playing Field with No Additional Bandwidth
- This is a Nonintrusive Method of Adding Uniform Munition Models to Large Number Of Simulations
- Independent Munition Models Simplify Verification and Validation Process
- An Ordnance Server Has Been Demonstrated to Help Solve the "Fair Fight" Problem